Modern Statistical Computing

Tipps for Seminar #1

January 20, 2023

```
# Packages I used
library(eurostat)
library(readr)
library(dplyr)
library(tidyverse)
library(lubridate)
```

Downloading the Data

To make sure that everyone pulls the correct data when working on the assignment (which most of you did), I wanted you to use the following tables from Eurostat:

- population (tgs00096)
- population density (tgs00024)
- population changes (tgs00099)
- fertility rate (tgs00100)

Tidying the Data

Once you have downloaded the different tables, I would make sure that you "clean them", which e.g. includes removing variables that are not needed for the analysis, renaming variables such that later on it is clear what they mean.[1] Moreover, before you merge the datasets, make sure that they are in the same format. As some of you have already realized yesterday, the population change dataset is not directly ready to be merged with the others. Specifically, it is still in long format with id variables `indic_de`, `geo` and `time`.

```
popchanges_df <- get_eurostat("tgs00099")
```

```
Reading cache file /tmp/RtmpJapK32/eurostat/tgs00099_date_code_FF.rds
```

```
Table  tgs00099  read from cache file:  /tmp/RtmpJapK32/eurostat/tgs00099_date_code_FF.rds
```

```
popchanges_df
```

```
# A tibble: 11,252 x 4
   indic_de   geo   time       values
   <chr>      <chr> <date>      <dbl>
 1 CNMIGRATRT AT11  2009-01-01    6.2
 2 CNMIGRATRT AT12  2009-01-01    3.3
 3 CNMIGRATRT AT13  2009-01-01    5.4
 4 CNMIGRATRT AT21  2009-01-01   -0.9
 5 CNMIGRATRT AT22  2009-01-01    1.8
 6 CNMIGRATRT AT31  2009-01-01   -0.3
 7 CNMIGRATRT AT32  2009-01-01   -1.3
```

[1] As you have noticed, in the raw downloaded version, the key variable will always be called values.

```
 8 CNMIGRATRT AT33   2009-01-01    1.3
 9 CNMIGRATRT AT34   2009-01-01    0.4
10 CNMIGRATRT BE10   2009-01-01   10.9
# ... with 11,242 more rows
```

When you go to the table at Eurostat, it contains an explanation of what each rowname means.

To merge it to the other datasets (which are in `geo` and `time` long format), we first have to reshape the `popchanges_df` dataset (and then either overwrite the existing one, or create a new dataset), for instance as follows:

```
popchanges_df <- popchanges_df  %>%
  pivot_wider(id_cols = c(geo, time), names_from = indic_de,
              values_from = values)
```

Selecting Years and Countries

Many of you struggled with selecting regions from two countries. One way would be to merge in my help file that maps regions to countries directly, or alternatively recognize that the first two letters of the NUTS2-codes contain the country code (according to Eurostat).

The former would be something along the lines:

```
# Read in help file (using your own filepath)
nutscountrymapping <- read_csv(file = file.path(datapath, "nuts2_overview.csv"))
```

```
Rows: 371 Columns: 3
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (3): nuts_code, countryname, nuts2_name

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# Merge with merged dataset (or alternatively, but less efficient with each individual table)

# For illustrative purposes, I will just merge with one table
# Important, in the nutscountrymapping dataset, the geographic variable is called nuts_code (and NOT GEO)
popchanges_df %>%
  left_join(nutscountrymapping, by = c("geo" = "nuts_code")) %>%
  filter(countryname %in% c("Spain", "Germany"))
```

```
# A tibble: 680 x 7
   geo   time       CNMIGRATRT GROWRT NATGROWRT countryname nuts2_name
   <chr> <date>          <dbl>  <dbl>     <dbl> <chr>       <chr>
 1 DE11  2009-01-01       -1     -1.4      -0.4 Germany     Stuttgart
 2 DE12  2009-01-01        1.7    0.1      -1.5 Germany     Karlsruhe
 3 DE13  2009-01-01        1      0.2      -0.8 Germany     Freiburg
 4 DE14  2009-01-01        0.3    0.1      -0.2 Germany     Tübingen
 5 DE21  2009-01-01        2.2    2.6       0.4 Germany     Oberbayern
 6 DE22  2009-01-01        0.3   -2.3      -2.6 Germany     Niederbayern
```

```
 7 DE23  2009-01-01        0.2   -2.2      -2.4 Germany     Oberpfalz
 8 DE24  2009-01-01       -1.2   -5.7      -4.5 Germany     Oberfranken
 9 DE25  2009-01-01        0.7   -1.3      -2.1 Germany     Mittelfranken
10 DE26  2009-01-01       -1.6   -4.2      -2.6 Germany     Unterfranken
# ... with 670 more rows
```

To select years, I would suggest using the package `lubridate` to make a year variable from the date format, and then simply filter for the year range 2011-2019.

Good luck with the rest of the assignment!